

2 4 02 00

## Sertifikaat

REPUBLIEK VAN SUID-AFRIKA

IB99/1389  
09/868270  
Certificate

PATENTKANTOOR



REC'D 28 FEB 2000

PATENT OFFICE

DEPARTEMENT VAN HANDEL  
EN NYWERHEID

REPUBLIC OF SOUTH AFRICA

DEPARTMENT OF TRADE  
AND INDUSTRYHiermee word gesertifiseer dat  
This is to certify that

the documents annexed hereto are true copies of:

REC'D 28 FEB 2000

WIPO PCT

Application forms P.1 and P.3, provisional specification and drawings of South African Patent Application No. 98/11657 as originally filed in the Republic of South Africa on 18 December 1998 in the name of GRAHAM PAUL GORDON for an invention entitled: "A METHOD OF PERFORMING A SYSTEM REVERSE ENGINEERING PROCESS";


AND it is further certified that Patent Application No. 98/11657 and the invention forming the subject matter of the patent application, together with all priority rights flowing from the patent application under the provisions of the International Convention were duly assigned in accordance with law by virtue of Deeds of Assignment from GRAHAM PAUL GORDON to THE NODROGO TRUST and from THE NODROGO TRUST to BUSHFIRE (PROPRIETARY) LIMITED, which Deeds of Assignment both were duly registered at the Patent Office, Pretoria, on 1 March 1999.

Geteken te  
Signed at

PRETORIA

in die Republiek van Suid-Afrika, hierdie  
in the Republic of South Africa, this14th dag van  
day of

February 2000

PRIORITY  
DOCUMENTSUBMITTED OR TRANSMITTED IN  
COMPLIANCE WITH RULE 17.1(a) OR (b)
  
 .....  
 Registrateur van Patente  
 Registrar of Patents

REPUBLIC OF SOUTH AFRICA  
PATENTS ACT, 1978  
APPLICATION FOR A PATENT AND  
KNOWLEDGEMENT OF RECEIPT  
(Section 30(1) Regulation 22)

(to be

M.P.1  
d in duplicate)

18.12.98

R 060.00

THE GRANT OF A PATENT IS HEREBY REQUESTED BY THE UNDERMENTIONED APPLICANT  
ON THE BASIS OF THE PRESENT APPLICATION FILED IN DUPLICATE

21 01 PATENT APPLICATION NO 9 11657

A&A REF V13178 MR

71 FULL NAME(S) OF APPLICANT(S)

GRAHAM PAUL GORDON

*The Nodroge Trust 1-3-99*

DISCLOSURE AND  
ALLOANIS & INSTITUTED

*Bushfire Pty Ltd 1-3-99*

ADDRESS(ES) OF APPLICANT(S)

519 MOORE ROAD, GLENWOOD, 4001, REPUBLIC OF SOUTH AFRICA

REGISTRAR OF PATENTS, DESIGNS, TRADE MARKS AND COPYRIGHT
PRIVATE BAG PRIVAATSAK X400
1998 -12- 1 8
PRETORIA 0001
REGISTRATEUR VAN PATENTE, MODELLE HANDELSMERKE EN OUTEURSREG

54 TITLE OF INVENTION

"A METHOD OF PERFORMING A SYSTEM REVERSE ENGINEERING PROCESS"

Only the items marked with an "X" in the blocks below are applicable.

☐ THE APPLICANT CLAIMS PRIORITY AS SET OUT ON THE ACCOMPANYING FORM P.2. The earliest priority claimed is

Country:

No:

Date:

☐ THE APPLICATION IS FOR A PATENT OF ADDITION TO PATENT APPLICATION NO 21 01

☐ THIS APPLICATION IS A FRESH APPLICATION IN TERMS OF SECTION 37 AND BASED ON  
APPLICATION NO 21 01

THIS APPLICATION IS ACCOMPANIED BY:

- ☒ A single copy of a provisional specification of 17 pages
- ☒ Drawings of 15 sheets
- ☐ Publication particulars and abstract (Form P.8 in duplicate) (for complete only)
- ☐ A copy of Figure of the drawings (if any) for the abstract (for complete only)
- ☐ An assignment of invention
- ☐ Certified priority document(s). (State quantity)
- ☐ Translation of the priority document(s)
- ☐ An assignment of priority rights
- ☐ A copy of Form P.2 and the specification of RSA Patent Application No 21 01
- ☒ Form P.2 in duplicate
- ☒ A declaration and power of attorney on Form P.3
- ☐ Request for ante-dating on Form P.4
- ☐ Request for classification on Form P.9
- ☐ Request for delay of acceptance on Form P.4
- ☒ Copy of Form P.1

74 ADDRESS FOR SERVICE: Adams & Adams, Pretoria

Dated this 17 day of December 1998

*[Signature]*  
M ROTTEVEEL

ADAMS & ADAMS  
APPLICANTS PATENT ATTORNEYS

The duplicate will be returned to the applicant's address for service as  
proof of lodging but is not valid unless endorsed with official stamp

REGISTRAR OF PATENTS, DESIGNS, TRADE MARKS AND COPYRIGHT
PRIVATE BAG PRIVAATSAK X400
1998 -12- 1 8
PRETORIA 0001
REGISTRATEUR VAN PATENTE, MODELLE HANDELSMERKE EN OUTEURSREG
REGISTRAR OF PATENTS

00100

PATENT APPLICATION NO		
21	01	9 11657

A&A REF: V13178 MR

LODGING DATE	
22	18 DECEMBER 1998

FULL NAME(S) OF APPLICANT(S)	
71	GRAHAM PAUL GORDON

FULL NAME(S) OF INVENTOR(S)	
72	GRAHAM PAUL GORDON

EARLIEST PRIORITY CLAIMED	COUNTRY	NUMBER	DATE
	33	NIL	31
			32
			NIL

NOTE: The country must be indicated by its International Abbreviation - see schedule 4 of the Regulations

TITLE OF INVENTION	
54	"A METHOD OF PERFORMING A SYSTEM REVERSE ENGINEERING PROCESS"

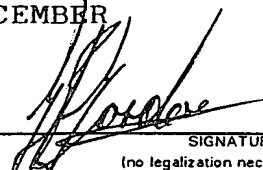
\* I/We GRAHAM PAUL GORDON

hereby declare that :-

1998-12-18
REGISTRAR OF PATENTS, DESIGNS, TRADE MARKS AND PATENT AGENTS HARBOUR ROAD, DURBAN 4001

- \* 1. I/we am/are the applicant(s) mentioned above;
- \*\* 2. I/we have been authorized by the applicant(s) to make this declaration and have knowledge of the facts herein stated in the capacity of \_\_\_\_\_ of the applicant(s);
- \*\*\* 3. the inventor(s) of the abovementioned invention is/are the person(s) named above and the applicant(s) has/have acquired the right to apply by virtue of an assignment from the inventor(s);
- 4. to the best of my/our knowledge and belief, if a patent is granted on the application, there will be no lawful ground for the revocation of the patent;
- \*\*\*\* 5. this is a convention application and the earliest application from which priority is claimed as set out above is the first application in a convention country in respect of the invention claimed in any of the claims; and
- 6. the partners and qualified staff of the firm of ADAMS & ADAMS, patent attorneys, are authorised, jointly and severally, with powers of substitution and revocation, to represent the applicant(s) in this application and to be the address for service of the applicant(s) while the application is pending and after a patent has been granted on the application.

SIGNED AT DURBAN THIS 15th DAY OF DECEMBER 1998

  
SIGNATURE(S)  
(no legalization necessary)

- \* In the case of application in the name of a company, partnership or firm, give full names of signatory/signatories, delete paragraph 1, and enter capacity of each signatory in paragraph 2.
- \*\* If the applicant is a natural person, delete paragraph 2.
- \*\*\* If the right to apply is not by virtue of an assignment from the inventor(s), delete "an assignment from the inventor(s)" and give details of acquisition of right.
- \*\*\*\* For non-convention applications, delete paragraph 5.

ADAMS & ADAMS  
PATENT ATTORNEYS  
PRETORIA

REPUBLIC OF SOUTH AFRICA  
Patents Act, 1978

## PROVISIONAL SPECIFICATION

(Section 30 (1) - Regulation 27)

21	01	OFFICIAL APPLICATION NO
----	----	-------------------------

9311657

22	LODGING DATE
----	--------------

18 DECEMBER 1998

71	FULL NAME(S) OF APPLICANT(S)
----	------------------------------

GRAHAM PAUL GORDON

ANDREW J. GORDON AND  
APPLICANTS

*The Nedrege Trust 1-3-99*  
*Bushfire Pty Ltd 1-3-99*

72	FULL NAME(S) OF INVENTOR(S)
----	-----------------------------

GRAHAM PAUL GORDON

54	TITLE OF INVENTION
----	--------------------

"A METHOD OF PERFORMING A SYSTEM REVERSE ENGINEERING PROCESS"

THIS INVENTION relates to a method of performing a system reverse engineering process.

It is known that as a result of software system accretion, which occurs when systems are linked together, when systems are built on or hacked into, when systems are modified to accommodate other systems and/or the like, a conglomerate system can result which, for various reasons, can no longer be easily managed and which is not understood in all respects. In extreme cases this can result in a system becoming obsolete, inutile, or too complicated to continue to operate and work with, essentially requiring system replacement.

The latter option often is not economically or technically feasible and in order to at least alleviate the problem identified, system reverse engineering processes

have been developed whereby conglomerated systems can be re-engineered into a workable format. System reverse engineering processes involve essentially the examination of the existing system, the documentation of the system, modelling of the system, analyzing of the system and understanding of the system, whereafter it is possible to re-engineer the system into a workable and useful format.

A system that requires to be reverse engineered as herein envisaged, hereinafter referred to as the application system, comprises a network structure of nodes and links, the nodes and links forming chains that either terminate in a final node or that form a closed loop that extends from a node and returns to the same node. Network structures are further complicated insofar as two or more links can extend from a node and by keeping in mind that a network structure could include millions of nodes and links, it will be appreciated that very intricate structures can result. The individual nodes and links referred to essentially are object instances representing activity or data elements which are associated with the operation of the system, for carrying out its required purpose or purposes.

In order to apply a reverse engineering process to an application system, it is required to obtain a full understanding of the system, i.e. an understanding of the operation of the network structure forming the system, in order to permit the system to be reverse engineered into a format which permits a model of the

system to be created with the aid of a suitable Case Tool (Computer Aided Software/Systems Engineering Tool). The examination of an application system in order to acquire an understanding of the system conventionally involves an overall consideration of the system and then progressively delving into the system from a number of predetermined starting points, delving deeper and deeper into the system until the required understanding is acquired. This generally requires the cooperation of a team of suitably qualified systems engineers who will cooperate with one another and add their knowledge together until the required level of understanding of the system is acquired, which then permits reverse engineering. This examination system is well known and produced desired results in relation to relatively simpler application systems where systems engineers could acquire a sufficient overall picture of the system to permit reverse engineering thereof, but in relation to more complex systems this method of examination became too complex and, as such, impractical.

The more complex application systems requiring reverse engineering therefore cannot be salvaged, even with the aid of software programs assisting with the examination process as above envisaged. As such, it is an object of this invention to provide an improved method of performing a system reverse engineering process which will permit more complex application systems to be reverse engineered to a workable format.

According to the invention there is provided a method of performing a system reverse engineering process, which includes the steps of

identifying the application system that requires reverse engineering and gathering the entire system and identifying the development environment to be associated with the system;

identifying initial object types that can serve as starting points from where an examination of the system must be initiated and analyzing the nature, characteristics and properties of each object type;

identifying entry points for entering the system to carry out the examination of the system;

examining from selected entry points the network structure forming the application system by tracking chains of nodes and links, each chain being tracked until the instance of a node that does not have a link or the return of the chain to a previously examined node, then reverse tracking the chain to a node from which another chain extends and selectively tracking said other chain and continuing the process until all the chains within the network structure have been tracked, the tracking of the chains including an examination of each node and link and a recordal of information so gathered; and



from the information gathered by the network examination, formatting the information gathered into a form in which it represents the application system in a usable form.

In identifying the application system that requires reverse engineering and gathering the entire system, it must be ensured that all the important components of the system are taken account of in order to ensure the effectiveness of the reverse engineering process. The development environment identified may be an integrated development environment which may include amongst others the program language and syntax used, the mechanisms of storage of data and the interface of the above.

The object types identified typically may fall into three categories or groups, namely process or activity structure elements, data structure elements and interface structure elements. The object types within these groups generally are comprised of nodes and links which form the overall network structure representing the application system.

The examination of the network structure comprises an examination of each node and each link in the structure to the extent that the nature, characteristics and properties of each node and link can be associated with an object type through analysis and understanding thereof, and all the relevant information of each node

and each link is then gathered and recorded. The entry points identified therefor may comprise the nodes from which a complete examination of the entire network structure of the application system can be initiated. The examination of the network structure also involves the complete tracking of each chain to its termination or return to an earlier node in the chain, before a further chain is selected and tracked.

The information gathered from the examination of the network structure will enable a complete understanding of the network structure and particularly also its object types, which in turn will permit formatting of this information into a logical format in which the application system is again rendered usable.

The method of the invention particularly provides for formatting of the information gathered into a format in which the information can be exported/reported to predetermined Case Tools, development environments and/or repositories, permitting the creation of a model of the application system. As such, formatting may include breaking structures into candidate components by using affinity analysis, mathematical cluster techniques, and the like.

The method of the invention provides still further for the employment of software for assisting with the identification of object types and the analysis of the nature, characteristics and properties of each object type identified, the identification of

entry points for entering the system to carry out the examination of the network structure forming the system, the actual examination of the network structure and the formatting of the information gathered by the examination of the network structure into a usable form.

As such, the method of the invention includes creating a software program that can be employed for the above purpose in respect of a particular application system.

Furthermore, in relation to the employment of the software program as a result of which object types are identified which were not originally accounted for, the method of the invention may include modifying the software program in order to take into account the object types so identified.

It will be understood in the above regard that although it is the employment of the software program that renders the method practically feasible, particularly in relation to more complex application systems, it is the method steps as defined and which must be followed, which renders the use of a software program for the purpose practically feasible.

The method of performing a system reverse engineering process, in accordance with the invention, can be used in respect of a wide range of application systems

that are associated with the problems hereinabove identified, the method of the invention essentially enabling these application systems to be salvaged by re-formatting of the systems through the reverse engineering thereof into a form in which the systems again be made practically usable.

The method of performing a system reverse engineering process, in accordance with the invention, is described hereinafter with reference to the accompanying diagrams. In the diagrams:

Figure 1 illustrates diagrammatically in block diagram form a flow chart illustrating the method of performing a system reverse engineering process, in accordance with the invention;

Figure 2 illustrates diagrammatically in block diagram form a flow chart setting out a non-application system process for illustrating the method of performing a system reverse engineering process, in accordance with the invention; and

Figure 3 (13 pages) illustrates diagrammatically in block form a particular example of a method of performing a system reverse engineering process, in accordance with the invention, referring by way of explanation to the block diagram illustrated in Figure 2.

Referring initially to Figure 1 of the drawings, a method of performing a system reverse engineering process, in accordance with the invention, is illustrated as a flow chart in block diagram form. Blocks 10, 12 and 14 represent the initiation of the method which includes the selection/identification of the application system that requires reverse engineering, the gathering of all the components of the application system which are required for the operation of the entire system and the identification/selection of the development environment associated with the system. This development environment may be an integrated development environment which includes the program language and syntax used, the mechanisms of storage of data and the interface of the above. Clearly, the development environment also may include other aspects which is associated directly with the application system involved.

Blocks 16 and 18 represent the method steps of identifying initial object types incorporated within the systems and that can serve as starting points from where an examination of the system must be initiated and analyzing and understanding the nature, characteristics and properties of each object type to enhance still further the overall understanding of the system. It must be appreciated in this regard that the object types identified essentially will fall into three categories or groups, i.e. a first group including process or activity structure elements, a second group including data structure elements and a third group including interface

structure elements. Examples of object types falling within the above three groups are set out below:

Examples of Process or Activity Structure Elements as Object Types (Active "things" or "things" that perform actions or do "things")

- Program
- Procedure
- Sub-Procedure
- Library Procedure
- Class/Method
- Call
- Invocation
- Message
- Command/Verb
- Statement
- Algorithm Flow/Control
- Rules/Conditions

Examples of Data Structure Elements (Passive "things" or "things" that have "things" done to them by "things" that do "things")

- Table
- Field
- File
- Entity
- Attribute
- Relationship
- Relation
- Array
- Variable
- Parameter
- Pointer

Examples of Interface Elements ("things" the user of the application system sees or interacts with.)

- Dialog
- Report

Screen (Read-only, Read-write)

Menu

Window

List

Button

Text Box

Check Box

Radio Button

Tree

It will be understood that additional object types falling within the above groups may exist within an application system and, as set out hereafter, these object types, once identified through the method of reverse engineering as hereinafter described, can then be categorized on an ad-hoc basis. It must also be understood at this stage that the object types referred to above generally are comprised of nodes and links which form the overall network structure representing the application system in respect of which reverse engineering is required, the nodes and links forming chains which themselves define the network structure.

The next step in the method of performing a system reverse engineering process is represented by block 20 and involves the identification of logical entry points through which the system can be entered for examination purposes.

Once these entry points have been identified, the next step within the method of the invention involves the examination of the network structure forming the

system, which includes selecting entry points from the logical entry points already identified and tracking the chains of nodes and links extending from these entry points, each chain being tracked until the instance of a node that does not have a link or the return of the chain to a previously examined node. This is then followed by a reverse tracking of the chain to a node from which another chain extends, selectively tracking the said other chain and continuing the process in the manner defined until all the chains within the network structure have been tracked, the tracking of the chains including also an examination of each node and link and a recordal of information gathered from this examination of each node and link. It must be understood that when reverse tracking of a chain is referred to, a reverse path along a chain will be followed until a node is identified from which another chain extends and a decision will then be made whether this chain requires tracking. If not, the reverse tracking will continue until a node is identified from which a chain extends which will again require tracking, the overall objective remaining that all the chains within a network structure and particularly all the nodes and links within the structure, must be examined and information in respect thereof must be gathered. In Figure 1 the blocks 22, 24 and 26 represent this examination process at the completion of which a complete understanding of the original application system should be possible insofar as all the nodes and links forming chains within the network structure representing the system will have been examined and the properties, functions and characteristics of the nodes and links will be known.



The final step in the reverse engineering method of the invention hence involves formatting of the information gathered by the examination referred to above, particularly using the information gathered for formatting the application system in a network structure form which is effectively usable, i.e. in a form in which the application system is understandable and the system can again be used for fulfilling its required purpose in a normal manner, while also permitting the application system to be worked with and modified as may be required from time to time. This latter step in the method of forming a system reverse engineering process is represented by the blocks 28 and 30 from which it will be appreciated that the newly formatted application system will be in a form in which predetermined case tools can be utilised for creating a model of the application system, if required.

In order to facilitate the application of the method of performing a system reverse engineering process, in accordance with the invention, the method steps associated with the blocks 16 to 30 can be carried out with the aid of a suitable software program that has been created for the purpose and particularly for use in conjunction with the application system to be reverse engineered. The creation of this software program accordingly also may form a part of the method of the invention, it being envisaged in this regard that it may be required to modify the software program from time to time as a result of unknown object types being identified during the examination process, permitting a systems engineer to

categorise the object type and then provide the necessary information within the software program in order to deal with this object type in carrying out the method of the invention as described. In this regard it will be appreciated that the method of the invention could be "manually" carried out in relation to relatively simple application systems, but in relation to practical application systems which do in fact require reverse engineering, the assistance of a suitable software program will be essentially required.

Referring to Figure 2 of the drawings, in order to explain the method of performing a system reverse engineering process, the process can be equated to the examination of a building having a plurality of rooms that are interlinked with one another in an essentially random fashion via doors between them and where the layout of the building is not known and therefore requires examination. In this diagram, block 40 can be associated with the identification of the application system to be reverse engineered, while block 42 provides for the identification of entrance doors which can lead into the system/building for examination purposes. Different selected entrance doors will then be entered by different members of the examination team involved.

The examination of the rooms within the building as represented by blocks 44, 46, 48, 50, 52, 54, 56, 58, 60, 62 and 64, will effectively represent the examination process associated with the method of the invention insofar as rooms

will be examined one after the other, until a room is reached which does not have a further door therein, following which the chain followed will be reversed until a room is reached from where another door extends and from where the examination process can continue. The process as described by the blocks 44 to 64 clearly will result in each room in the building being examined, which will in fact equate to the examination of all the nodes and links within a network structure of an application system.

All the information gathered from the individual rooms will then be set out in a logical format, which operation is represented by the block 66, this logical format enabling one to obtain a clear picture of the layout of the building involved. This layout of the building, which will now be clear, will equate to the new format of the application system that has been created, which will comprise an understandable format which will give a clear understanding of the application system and, particularly, the network structure forming the system.

This new format insofar as it applies to the building examined clearly will then permit additions to the building to be effected or a model of the building to be built and this clearly equates to the use of the new format in relation to the method of the invention which permits the creation of a model of the application system involved with the aid of suitable case tools, or merely the normal application of the application system which may require system modifications,

additions and the like, which could again be logically carried out as a result of the complete understanding of the system which is acquired through the reverse engineering process as described.

Referring to Figure 3 of the drawings, a typical example of a comprehensive automated reverse engineering method which includes the employment of the method of the invention is described in a block diagram form and in association with the simulation of the method of the invention as illustrated in Figure 2 of the drawings. The individual steps as illustrated in Figure 3 of the drawings therefore are cross-referenced as step numbers with reference to Figure 2 of the drawings, the step numbers being associated with the numbers 1 to 10 included within the blocks forming the diagram illustrated in Figure 2 of the drawings. As the method of the invention in its application with reference to Figure 3 will be clear to those skilled in the art, the method as illustrated in Figure 3 of the drawings is not described in detail hereafter.

It must be appreciated that the method of the invention can be applied in association with many different application systems that require reverse engineering, essentially enabling salvaging of application systems which may otherwise have become obsolete, insofar as the application systems will be formatted into forms in which the systems are again rendered usable through the effective understanding of the systems.

DATED THIS 17th DAY OF DECEMBER 1998



ADAMS & ADAMS  
APPLICANTS PATENT ATTORNEYS

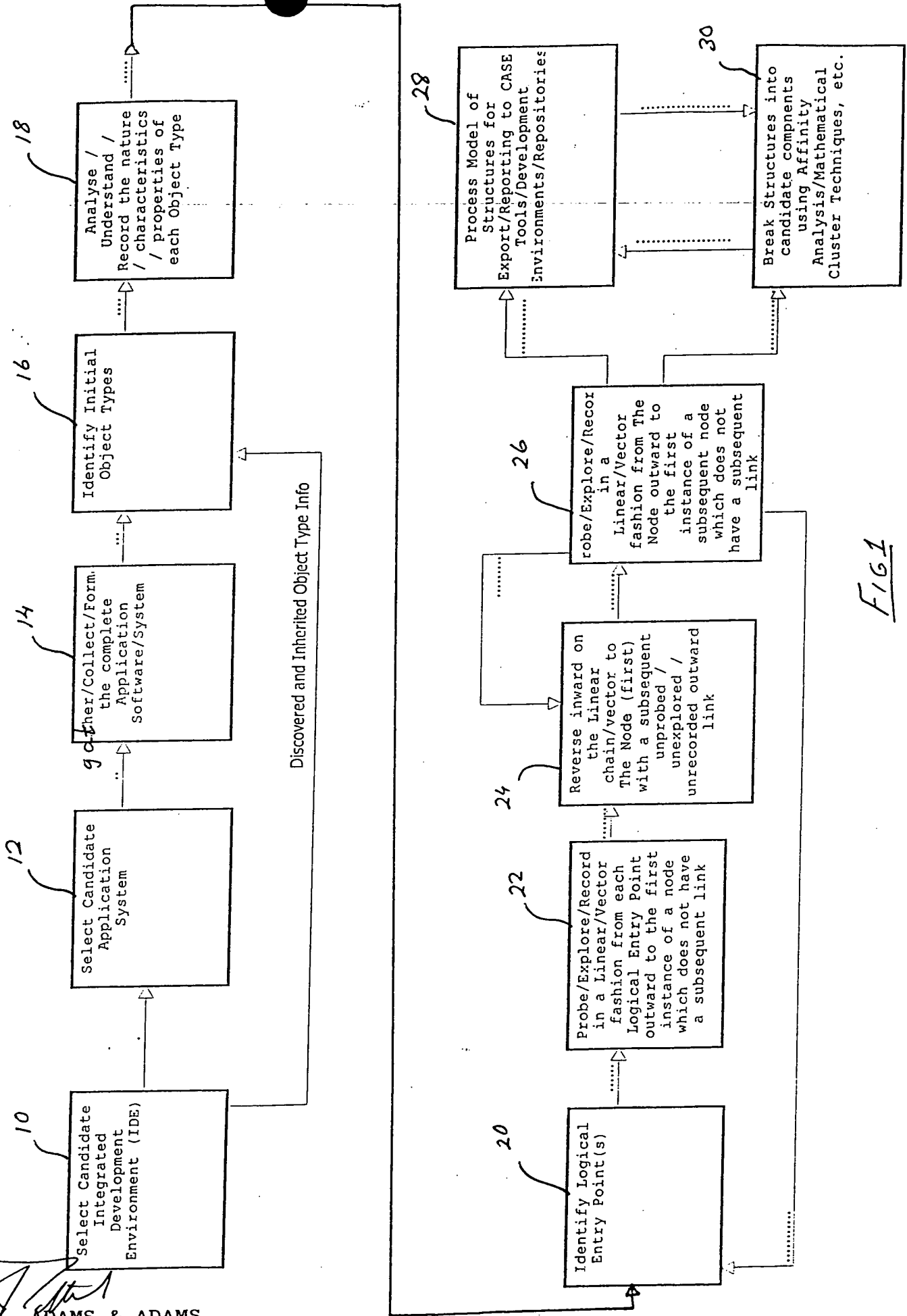
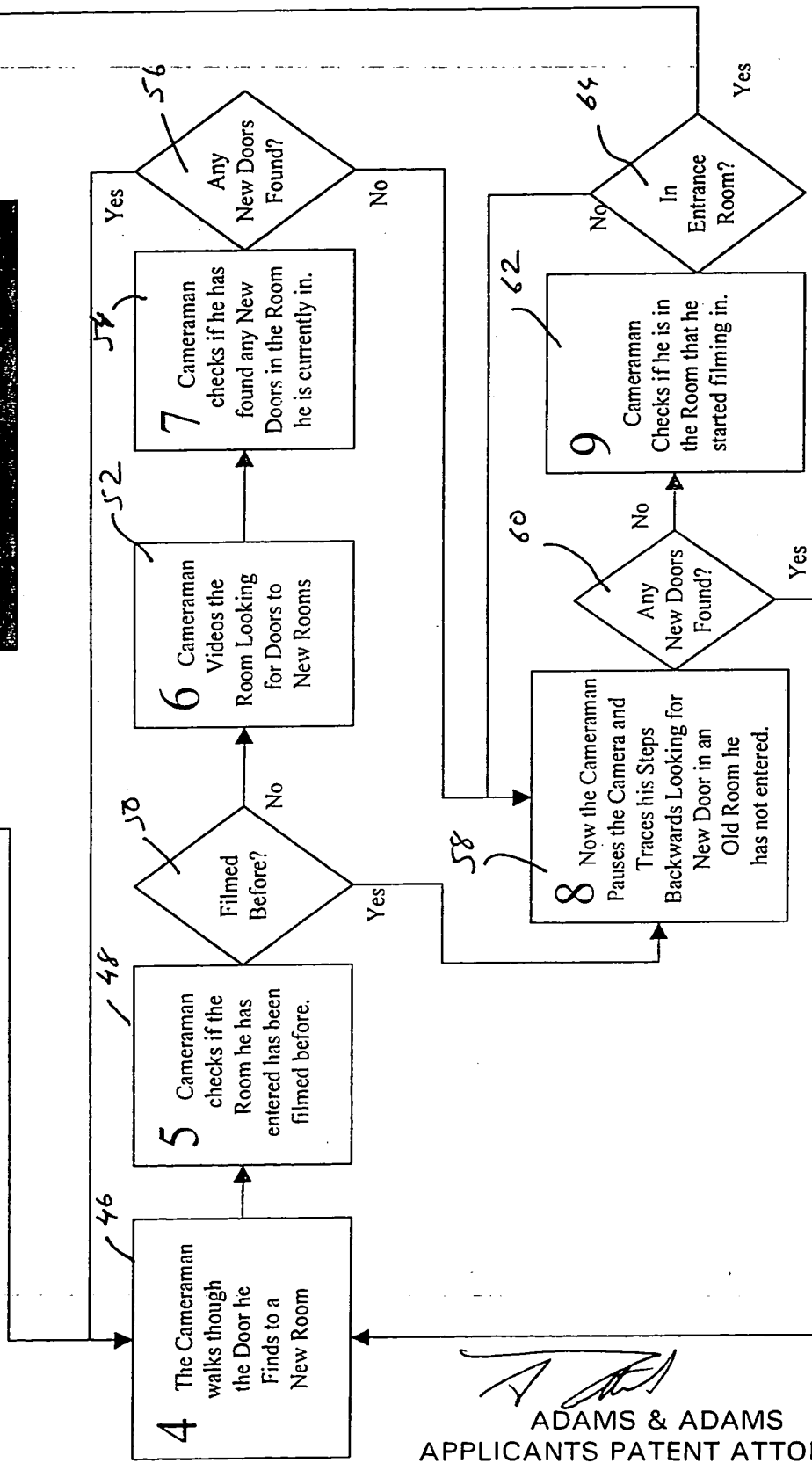
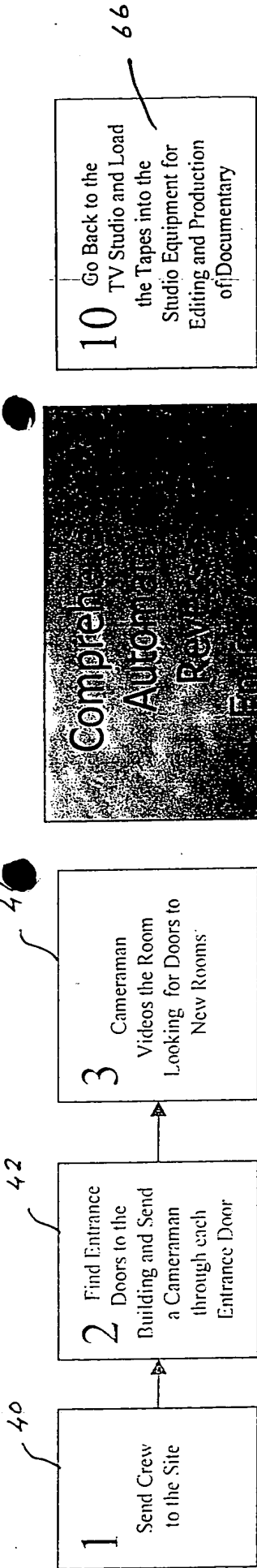


FIG 1



ADAMS & ADAMS  
APPLICANTS PATENT ATTORNEYS

FIG 2

## Comprehensive Automated Reverse Engineering Process / Method Example

Limited to a single entry point of a  
SQL Integrated Development  
Environment.

*SQL Data Definition  
"Create Table" statement*

FIG 3A

## Overview

- This example uses a non-systems oriented example (Crazy Filming Co.)
- The example steps are cross referenced against an extract of a Comprehensive Automated Reverse Engineering exercise on a subset of a working Application System.

FIG 3B

FIG 3 (13 pages)

## Identify an Application System

- An application system representing a complete business application must be located. **Step 1**
- This will be the source system for Comprehensive Automated Reverse Engineering. **Step 1**

FIG 3C

## Gather all input files into VIRTUAL file (memory)

- For each physical input file **Step 2**
  - Read all lines from the input file into the VIRTUAL file **Step 2**

FIG 3D



## Analyse, Parse and Record all Metadata in VIRTUAL file

- For each line in the VIRTUAL file, we iterate **Step 3, 8 & 9**
  - Assign the current line to storage variable
  - If line has "create table" then
    - Start looking for the "end of create table" marker by starting a new iteration, from the current position in the list. **Step 3**
    - Read the first of the new set of lines and reformat to ensure we have no unnecessary characters (tab stops and double spaces) in the line. **Step 3**

FIG 3E

## Analyse, Parse and Record all Metadata in VIRTUAL file

- While the current line does not contain the "end of table" marker then: **Step 4**
  - Concatenate the current line with the storage variable **Step 4**
- Concatenate the current line, which contains the "end of table" marker, with the storage variable. **Step 4**
- If format of "create table" statement is correct then **Step 4**
  - Strip the table name from the table definition, and store each as a substring. **Step 4**

FIG 3F

## Analyse, Parse and Record all Metadata in VIRTUAL file

- Check if the table item exists in our nodal model, if it does not then: **Step 5**
  - Add the table name to the list of tables with a unique ID. **Step 6**
  - Trim known garbage from the ends of the table definition. **Step 7**
  - Find fields within the table definition. **Step 7**

Fig 35

## Analyse, Parse and Record all Metadata in VIRTUAL file

- For each of the fields in the table definition line  
**Step 4 & 7**
  - Split the field into field name, field type, field storage format and field data entry constraint. **Step 4**
  - Check if it exists as nodal item linked to this table item, if it does not, then: **Step 5**
    - > Add the field to the list of fields with a unique ID. (Record) **Step 6**
    - > Add a tablefield link to the list of links for this field and current table. **Step 6**

Fig 34

## Analyse, Parse and Record all Metadata in VIRTUAL file

- Else if format of "create table" incorrect then  
Step 4 & 7
  - Identify the incorrect statements line number, file name and location of file. Step 4
  - Check if this Unknown item exists as nodal item linked to the current file, if it does not, then:  
Step 5
    - Add the incorrect statement to the list of "Unknown" items with a unique ID Step 6
    - Add an unknownfile link to the list of links for this Unknown and respective file Step 6

FIG 3I

## Analyse, Parse and Record all Metadata in VIRTUAL file

- End of has "create table" condition. Step 8 & 9
- If line has "create index" then
  - Check validity of create index statement. If it's invalid,
    - Identify the incorrect statements line number, file name and location of file. Step 4
    - Check if this Unknown item exists as nodal item linked to the current file, if it does not, then:  
Step 5
      - Add the incorrect statement to the list of "Unknown" items with a unique ID Step 6
      - Add a unknownfile link to the list of links for this Unknown and respective file Step 6

FIG 3J

## Analyse, Parse and Record all Metadata in VIRTUAL file

- If the create index statement is valid:
  - Identify the field that the index is being created on. **Step 4**
  - Check if this "create index" item exists as nodal item linked to the identified field, if it does not exist: **Step 5**
    - Add the "create index" item to the nodal item list of indexes with a unique identifier. **Step 6**
    - If the field being indexed does not yet exist, add the new "create index" item to the list of items pending linkage. **Step 6**
    - Else Create an indexField link to the list of links that connects this "create index" to the affected field. **Step 6**

FIG 3K

## Analyse, Parse and Record all Metadata in VIRTUAL file

- If the "create index" statement exists: **Step 5**
  - Add the incorrect or duplicate statement to the list of "Unknown" items with a unique ID **Step 6**
  - Add a unknownindex link to the list of links for this Unknown and its respective field. **Step 6**
- End of VIRTUAL file iteration. **Step 10**

FIG 3L

## Analyse, Parse and Record all Metadata in VIRTUAL file

- Iterate through list of items pending linkage. **Step 7**
  - If an the current unlinked item now has valid links:
    - create an item specific link item in the link list. **Step 6**
  - If the current unlinked item still has invalid link points:
    - Add an unknown item to the list of unknown items with a unique identifier. **Step 6**
    - Add a link from this unknown item to the file the unlinked item occurred in, and the line in that file. If there is a single side to the link, add an unknown item link from this unsuccessful link object to the known link point. **Step 6**
- End of items pending linkage iteration.

FIG 3M

## Actual Data Manipulation Stages

FIG 3N

## Original Statement In SQL Definition File

```
create table cheqmast
(
    cheque no integer,
    cheque value decimal(9,2) not null,
    date chque date,
    practice char(7),
    delind char(1)
);
create index i cheqmast on cheqmast (
    cheque no );
```

Fig 30

## After Concatenation and Whitespace Cleaning

```
create table cheqmast (cheque no
integer,cheque value decimal (9,2) not
null,date chque date,
practice char (7),delind char (1)
);
```

Validity is tested at this point.

Fig 3P

## After Stage One Reformatting

(No space before comma)

```
create table cheqmast(cheque no
integer,cheque value decimal(9,2) not
null,date chque date,
practice char(7),delind char(1)
);
```

FIG 3Q

## After Stage Two Reformatting

(Single space after open bracket)

```
create table cheqmast( cheque no
integer,cheque value decimal( 9,2) not
null,date chque date,
practice char( 7),delind char( 1)
);
```

FIG 3R

## After Stage Three Reformatting

(Single space after open comma)

```
create table cheqmast( cheque no integer,  
    cheque value decimal( 9, 2) not null,  
    date chque date,  
    practice char( 7), delind char( 1)  
);
```

F1635

## After Stage Four Reformatting

(Single space before closing bracket)

```
create table cheqmast( cheque no integer,  
    cheque value decimal( 9, 2 ) not null,  
    date chque date,  
    practice char( 7 ), delind char( 1 )  
);
```

F1635



## Split Table Name and Definition

```
create table cheqmast( cheque no integer,  
cheque value  
decimal( 9, 2 ) not null, date chque date,  
practice char( 7 ), delind char( 1 )  
);
```

And leaves us with:

Table name : cheqmast

Table definition : ( cheque\_no integer, cheque\_value  
decimal( 9, 2 ) not null, date\_chque date,  
practice char( 7 ), delind char( 1 )  
);

The table node is stored.

FIG 34

## After Trim known garbage from the ends of the definition

```
cheque no integer, cheque value decimal( 9, 2  
 ) not null, date chque date,  
practice char( 7 ), delind char( 1 )
```

FIG 35

## After Splitting Fields

```
cheque no integer
cheque value decimal( 9, 2 ) not null
date chque date
practice char( 7 )
delind char( 1 )
```

FIG 3W

## Find data types, formats and constraints, finds the following

```
cheque value decimal( 9, 2 ) not null
```

Name	Type	Format	Constraint
cheque_value	decimal	9, 2	not null

Which is stored as:

```
Field Name      : cheque_value
Field Type      : decimal
Field Storage Format : 9, 2
Field Data Entry Constraint : not null
```

Each of the field nodes are stored and linked, using a link node, to their respective table node and the next table node is found and analysed.

FIG 3X

## Analyse "create index" statement

```
create index i_cheqmast on cheqmast (cheque_no);  
^-----^ ^-----^ ^-----^
```

Statement	Name	Table	Field
-----------	------	-------	-------

Which is stored as:

Index Name	:	i_cheqmast
Index Table	:	cheqmast
Field to index on	:	cheque_no

If the table cheqmast does not yet exist, or does not yet have a field by the name cheque\_no, the "create index" object is created and placed in the "to link" list.

If the table and field both exist, the "create index" object is created and an index-field link is added to the list of links.

F163Y

**THIS PAGE BLANK (USPTO)**